CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Lecture 3: Quality Scenarios

Gregory Gay
DIT636/DAT560 - January 27, 2025

When is Software Ready for Release?

Software is ready for release when you can argue that it shows sufficient quality.

- Requires choosing **quality attributes**.
  - Requires specifying **measurements** and **thresholds**.
  - May require different measurements and thresholds for **different functionality and execution scenarios**.
- Assessed through **Verification and Validation**.

Quality Attributes

- Describe **desired properties** of the system.
- Developers prioritize attributes and design system that meets chosen thresholds.
- Most relevant for this course: **dependability**
  - Ability to *consistently* offer **correct** functionality, even under *unforeseen* or *unsafe* conditions.

Quality Attributes

- **Dependability**
  - Ability to offer correct functionality, even under unforeseen or unsafe conditions. Encompasses correctness, **reliability**, safety, robustness
- **Availability**
  - Ability to carry out a task when needed, to minimize “downtime”, and to recover from failures.
- **Performance**
  - Ability to meet timing requirements. When events occur, the system must respond quickly.
- **Scalability**
  - Ability to maintain dependability and performance as the number of concurrent requests grows.

Today's Goals

- We have discussed what "quality" can mean (attributes).
- We have discussed how to measure reliability, availability, performance, and scalability.
- Today: How to assess each using **scenarios**.

Assessing Quality

- Quality goals differ between parts of the system.
  - (e.g., user-facing functionality, code components).
- For those parts, required quality may differ:
  - Between situations.
    - (e.g., system under normal use, heavy load, partially down)
  - Between different groups of users.
    - (e.g., new versus experienced users)
- **Scenarios** describe different execution conditions.

Assessing Quality

- For each scenario, set quality **thresholds** (targets).
- Execute the scenario, measure quality, compare to the threshold.
  - If the threshold is met, good!
  - If not, there is a fault in the system.
- Is that enough?

Assessing Quality

- Quality is often non-deterministic.
  - Performance differs every time code is run.
- Quality evolves with the code.
- Scenarios must be executed many times.
  - Thresholds usually set based on averages or probabilities (“95% of the time…”).
  - Absolute thresholds when there must never be a violation.

Writing Scenarios

Scenarios

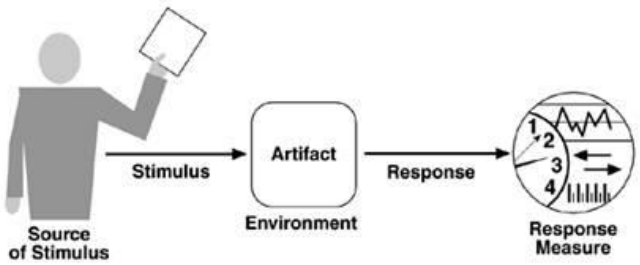
- Description of interaction between external entity and system. Defines:
  - Event that triggers the scenario.
  - Interaction initiated by the external entity.
  - Response required (in terms of quality attribute).
- Similar to use cases or user stories, but examines both quality **and** functionality.

Scenarios

Capture a range of requirements:

- A set of interactions with users to which a system must respond.
- Processing in response to timed events.
- Peak load situations that could occur.
- Regulator demands.
- Failure response.
- A change that a maintainer might make.
- Any situation that the design must handle.

Scenario Format



Overview:

Brief description of the scenario.

System State:

Aspects of internal state that affect quality (e,g., important information stored in the system, current load)

Environment State:

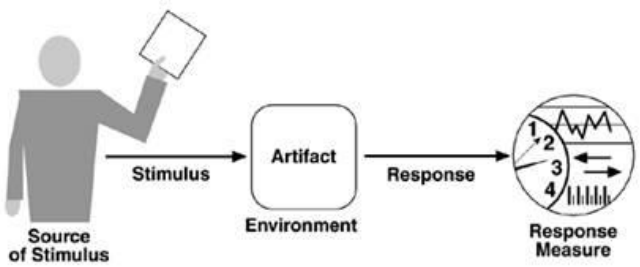
Observations about the environment (e.g., network connection, external system availability).

Scenario Format

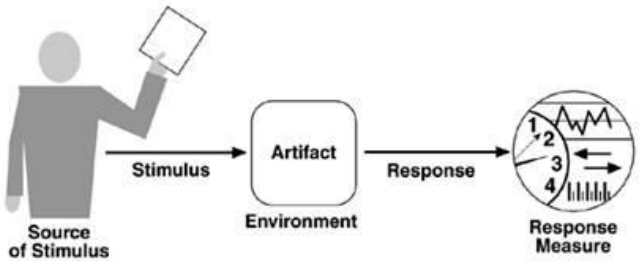
External Stimulus:

Event (input or environmental factor) that triggers a response from the system.

(e.g., user request, infrastructure changes or failures, attacks, expiration of time limit)



Scenario Format



Required System Response:

How does the system respond and meet the quality goal?

(e.g., how should it handle a defined increase in requests?)

Response Measure:

How we judge whether the system meets the quality goal.

(e.g., throughput, timing, availability)

Response Measures

- Many quality measurements are non-deterministic.
- May be OK if quality goals are violated *occasionally*.
  - **Most measures should be probabilistic.**
    - 95% of the time, the response should be N. **(common case)**
    - 99% of the time, the response should be M. **(worst case)**
    - If not non-deterministic, can give an absolute threshold.
  - When safety is affected, measurements should also give a **worst-case limit**.

Reliability Example

- **Overview:** How the system's end-of-day processing behaves when regular data volumes are suddenly greatly exceeded.
- **System State:** The system has stored summary statistics in its database for data that has been processed, and the system's processing elements are under light load (1,000 to 1,500 data items per hour).
- **Environment State:** The deployment environment is working correctly. The network connection is adequate.
- **External Stimulus:** The data update rate suddenly increases to 4,000 items/hour.
- **Required System Behavior:** When end-of-day processing starts, the system should process the data set until processing time exceeds a system-configurable limit. At that point, the system should stop, leave the summary statistics in place, and log a diagnostic message.
- **Response Measure:** Availability should remain at 99.999%.

Availability Example

- **Overview:** Client-facing web service fails during transmission of page update.
- **System State:** System is working correctly under normal load (10,000 concurrent users).
- **Environment State:** Deployment env. is operating normally. All services have adequate network connection.
- **External Stimulus:** Customer has generated a “add item to cart” post request, which was routed to Server <X> in transaction pool. <X> crashes during response generation.

Availability Example

- **Required System Behavior:** Load balancer should remove the service from the service pool after 2s of missed heartbeat messages, or on next request sent to the service. A page reload will be routed to alternate service by load balancer and the reloaded page should be correctly displayed on client-side.
- **Response Measure:** On client-side page refresh, client state and display contains state after last transaction. Time for re-routed refresh is equivalent to “standard” refresh (<1 second 95% of the time, <3 seconds 99% of the time).

Is a Scenario a Test Case?

- Scenarios prescribe general execution conditions and goals under those conditions.
  - Describe the “type” of event and response that occurs, not the specific input or output.
- Test cases require concrete, specific input.
- **One scenario** can be used to create **many concrete test cases**.

“Good” Scenarios

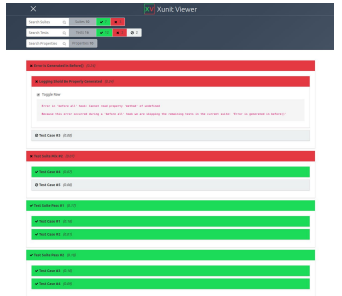
- Credible
  - Describes a realistic scenario.
- Valuable
  - Can be directly used to design and verify system.
- Specific
  - Addresses a single, concrete situation.
- Precise
  - User should be clear about the described situation and response.
- Comprehensible
  - Writing should be unambiguous and free of jargon.

Effective Scenario Use

- Identify a focused scenario set
  - Too many scenarios can be distracting.
  - Prioritize no more than 15-20.
- Use distinct scenarios
  - Avoid redundant scenarios.
  - Consider demands placed on the system.
- Use scenarios early
  - Can focus design activities.

What do we do with Scenarios?

- System Design
- Stakeholder Negotiation
- Exploratory Testing
  - Human experiments with app.
- Formal Test Cases
  - Assign specific input and check response.



Reliability Scenarios

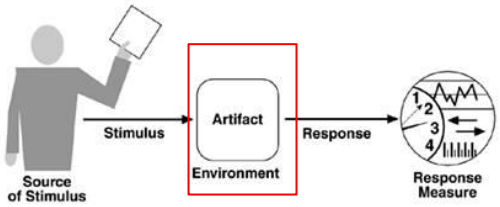
Reliability Scenarios

- **Ability to operate correctly over an observation period** (time or executions).
  - A statistical approximation of correctness.
- Scenarios revolve around one function/code component (or closely-related components).
  - Demonstrates that the function operates correctly.
  - Give context on type of user or system load if it impacts system execution or perceived reliability.

Reliability Metrics

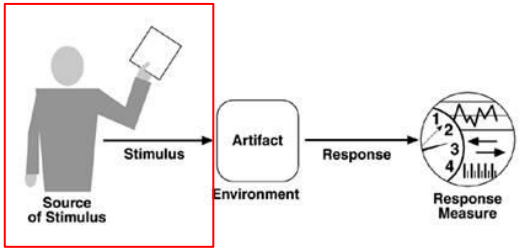
- POFOD: **(failures/ requests over period)**
- ROCOF: **(failures / total time observed)**
- Availability: **(uptime / total time observed)**
- MTBF: **Average time between observed failures.**

Reliability Scenarios



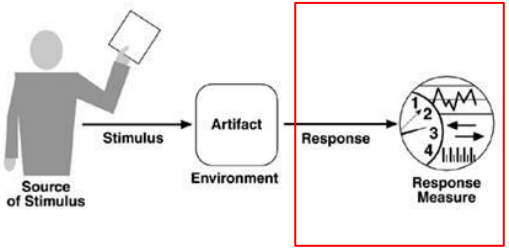
- **Overview:** Highlight the function(s) being used and the context used in. Explain the type of user/load, if relevant.
- **System State:** Data stored in system, past events (multiple failures may leave system vulnerable), load may impact reliability.
- **Environment State:** Available resources may impact reliability (resources, networking).

Reliability Scenarios



- **External Stimulus**:
  - User or external software performs one or more input interactions.
  - State specific interaction(s) performed.
  - If relevant, explain the type of user and reason they would perceive reliability differently.

Reliability Scenarios



- **Required Response:**
  - The functional response of the system.
  - Generally, the normal operation of that function.
- **Reliability Measure:**
  - ROCOF, POFOD, Availability, MTBF
  - Can be either absolute or average.
    - Every time scenario evaluated, ROCOF must be < 2 failures/day.
    - Across all evaluations, the average ROCOF must be < 2 failures/day.

Example Reliability Scenario

- **Overview**: A user adds an item to the shopping cart.
- **System State**: The system is operating under normal load (500 concurrent users). The shopping cart is empty.
- **Environment State**: The environment is operating normally, with standard network and database connectivity.
- **External Stimulus**: A user selects the “add to cart button” with the quantity set to “1”.
- **Required Response**: The item is successfully added to the shopping cart. The number of items displayed on the cart icon is incremented by one.
- **Response Measure**: The average ROCOF is 0.015 (15/1000 requests).

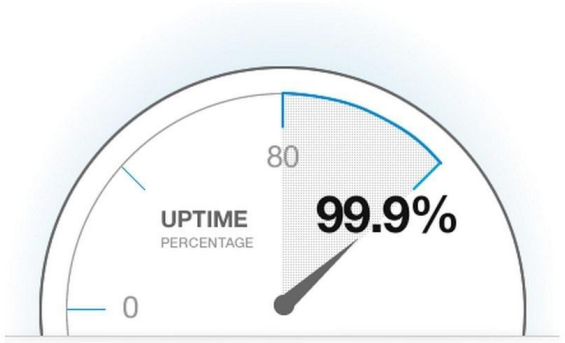
Example Reliability Scenario 2

- **Overview**: A “power user” requests summary statistics for data in the spreadsheet.
- **System State**: The system is operating under normal load (100 concurrent users).
- **Environment State**: The environment is operating normally, with adequate network connection.
- **External Stimulus**: A “power user” clicks the data summary button.
  - A power user manages a large volume of data (> 20000 rows), and accesses this function at least once per hour.

Example Reliability Scenario 2

- **Required Response**:
  - The summary statistics are calculated and displayed to the screen. The statistics are also written to a CSV file (appended to the file if it already exists).
- **Response Measure**:
  - The MTBF for this function must be at least 8 hours.
    - Power uses expect long, uninterrupted sessions, and expect accurate results on a regular basis.

Let’s take a break.



Availability Scenarios

Availability Scenarios

- Ability to recover from or work around failures in a reasonable manner and time period.
- Measure how the system **responds to failure**.
  - How is the failure detected?
  - What does the system do to return to normal?
  - How long does it take?
- **Stimuli should always be a failure**.

Availability vs Reliability

- Reliability scenarios:
  - How a function operates normally.
  - Measures show **how often it is allowed to fail**.
- Availability scenarios:
  - What happens when the function fails.
  - How system avoids failing or recovers from failure.
  - Response measures based on **successful avoidance or recovery time**.

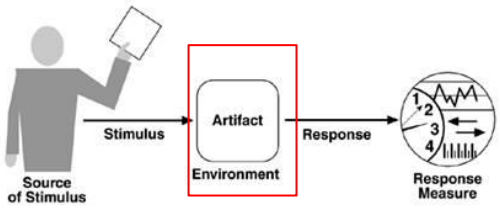
Availability Scenarios

- Response measures should always include a measure of availability:
  - availability threshold
    - (“Availability must be at least 0.9999”)
  - time to detect or repair fault
    - (“95% of the time, the failure is detected within 5ms”)
  - time system in degraded mode
    - (“95% of the time, must be back online within 10 minutes”)

Availability Scenarios

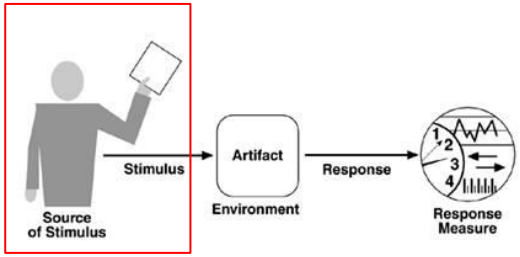
- Distinguish between failures and software's perception of the failure.
  - Do not assume software is omniscient.
- Scenarios tend to deal with:
  - Failure of internal component or external system.
  - Reconfiguration of physical system.
  - Maintenance or reconfiguration of software.

Availability Scenarios



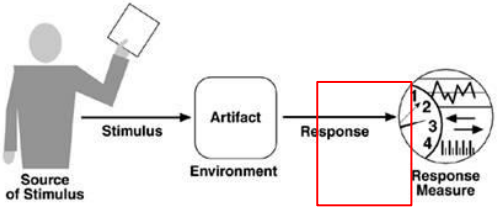
- **Overview:** Be clear about what failed and what needs to be available.
- **System/Environment State:** The state of the system when failure occurs may affect the response.
  - If this is the first failure, may choose degradation of response time or functionality.
  - If the system has already failed and is not in normal mode, may choose to shut it down.

Availability Scenarios



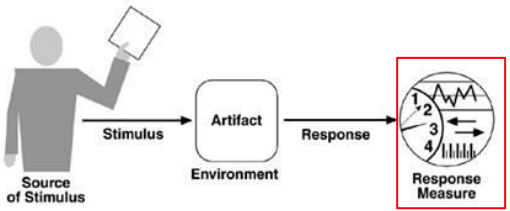
- **External Stimulus:** Differentiate internal/external failure - desired response may differ.
- Stimuli is:
  - An ***omission*** (component fails to respond)
  - A ***crash***
  - ***timing*** (component responds but is early/late)
  - ***incorrect response***

Availability Scenarios



- **Required Response:**
  Failure must be detected and isolated before recovery. Can:
  - Logging the failure
  - Notifying users or systems
  - Taking actions to limit the damage
  - Switching to a degraded mode
  - Shutting down external systems
  - Becoming unavailable during repair.

Availability Scenarios



- **Response Measure:**
  - Can specify an availability percentage
  - Can specify a time:
    - to detect the fault
    - to repair the fault
    - times where system must be available
    - duration system must be available

Example Availability Scenario

- **Overview:** How the server-side components handle non-response from external payment system.
- **System State:** System is operating under heavy load (>10000 concurrent users).
- **Environment State:** External payment processing system has exceeded load limits.
- **External Stimulus:** Multiple (5+) queries to the external system have gone without response without any successful responses between.

Example Availability Scenario

- **Required System Behavior:** The system will cease to allow any further orders until the external system responds to a heartbeat message. An error page will be displayed to all clients to prevent further order attempts. Once the external system has returned for sufficient time (100 responses over at least a five minute period), normal operations will be resumed.
- **Response Measure:** Following detection of the failure, all order attempts will be stopped within one minute (90% of the time) and two minutes (99% of the time). Once the failure is resolved, normal operations will be resumed within one minute (following the five minutes of successful responses) 90% of the time and three minutes 99% of the time.

Performance and Scalability Scenarios

Performance Scenarios

- Begin with events arriving at the system.
  - Responding requires resources to be consumed.
- Arrival pattern for events can be:
  - Periodic (at regular time intervals)
  - Stochastic (events arrive according to a distribution)
  - Sporadic (unknown timing, but known properties)
    - “No more than 600 per minute”
    - “At least 200 ms between arrival of two events”

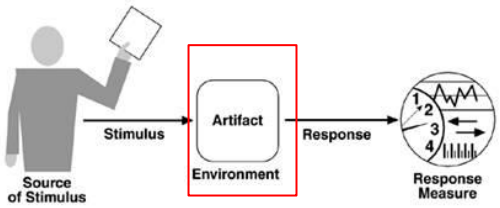
Performance Measurements

- **Latency:** The time between the arrival of the stimulus and the system's response to it.
- **Response Jitter:** The allowable variation in latency.
- **Throughput:** Number of transactions system can process in a unit of time.
- **Deadlines in processing:** Points where processing must have reached a particular stage.
- **Number of events not processed** because the system was too busy to respond.

Specifying Response Time

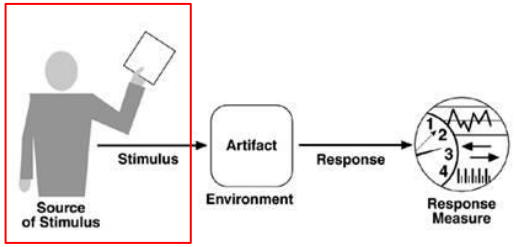
- Response time targets require a defined load.
  - One transaction in 3s is easy if that is the only request.
    - Can you still hit 3s if there are 500 transactions per second?
  - Specify a clearly-defined response time goal.
    - Define when a transaction starts and ends.
- Not all requests take the same amount of time, even with constant load.

Performance Scenarios



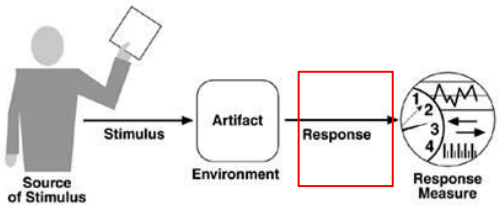
- **Overview:** Description of the scenario.
- **System State:** System can be in various levels of load (normal, emergency, peak load, or overload).
- **Environment State:** Be clear on conditions that can impact performance.
  - Limited resources (disc, memory, CPU)
  - Networking conditions

Performance Scenarios



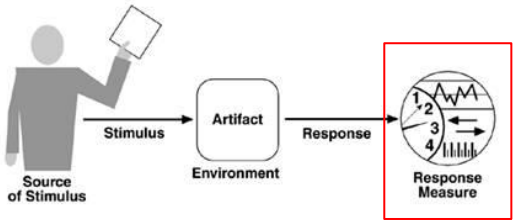
- **External Stimulus:** Stimuli arrive from external or internal sources.
  - The stimuli are event arrivals.
  - Arrival pattern can be periodic, stochastic, or sporadic, characterized by numeric parameters.
  - Be clear on number, duration, concurrency of stimuli.

Performance Scenarios



- **Required System Behavior:**
  - System must process arriving events.
  - May cause change in system
    - (e.g., shift from normal to overload mode).
  - May cause change in environment
    - (e.g., reduction of available memory)

Performance Scenarios



- **Response Measure**:
  - Time to process arriving events (**latency or a deadline**)
  - Variation in latency time (**jitter**)
  - Number of events that can be processed within a time interval (**throughput**)
  - Characterization of the events that cannot be processed (**miss rate**).

Example Performance Scenario

- **Overview:** Check responsiveness for adding items to shopping cart under normal conditions.
- **System State:** Normal load (less than 20 customer requests per second).
- **Environment State:** Sufficient internet connection to client. External dependencies are functioning.
- **External Stimulus:** Customer adds product to cart.

Example Performance Scenario

- **Required System Behavior:** Web page refreshes. Icon on right side of web page displays last item added to cart. If item is out of stock, cart icon has exclamation point overlay on top of cart icon.
- **Response Measure:** In 95% of requests, web page is loaded and displayed to user within 1 second. In 99.9% of requests, web page is loaded and displayed to user within 5 seconds.

Example Performance Scenario 2

- **Overview:** Ensure that credit card processing can still meet throughput targets when many users are competing for resources.
- **System State:** System is operating under heavy load (10000 concurrent users are logged in).
- **Environment State:** Load balancer distributes user requests approximately evenly between 100 servers.
- **External Stimulus:** A large number of credit card processing requests come within a short window (8500 requests within a one minute window).

Example Performance Scenario 2

- **Required System Behavior:** Each server maintains a queue of requests and processes requests as resources become available. The load balancer distributes requests to servers, favoring servers with shorter queues. All requests are completed successfully.
- **Response Measure:** All 8500 requests are completed within two minutes, 85% of the time. All requests are completed within three minutes 99% of the time.

Scalability Scenarios

- **The ability to effectively use available resources**
- Scenarios directly deal with impact of **adding or removing resources**.
- Performance measures to reflect:
  - Changes to performance.
  - Changes to availability.
  - Load assigned to existing and new resources.

Example Scalability Scenario

- **Overview:** Addition of new hardware improves credit card transaction speed.
- **System State:** Before addition of new hardware, 95% of credit card transactions were completed within 10 seconds, 99.9% within 15s. System is under normal load (1500 users).
- **Environment State:** An additional server has doubled threads available for processing requests. Environment is otherwise operating normally, with adequate connectivity.

Example Scalability Scenario

- **External Stimulus:** Customer completes a purchase.
- **Required System Behavior:** Order confirmation is displayed, with a list of items purchased, expected arrival date, and total cost of items.
- **Response Measure:** In 95% of requests, web page is loaded and displayed to user within 5 second. In 99.9% of requests, web page is loaded and displayed to user within 7.5 seconds.

Example Scalability Scenario 2

- **Overview:** Addition of additional VMs improves availability of account authorization service.
- **System State:** The average availability of the authorization service was previously 97.35% per week. System is under normal load (15,000 users).
- **Environment State:** The VM pool has now been increased by 150%. Environment is otherwise operating normally, with adequate connection.
- **External Stimulus:** A user submits their username and password for authentication. The password is correct.

Example Scalability Scenario 2

- **Required System Behavior**:
  - Authentication completes successfully.
  - A session is established, and the user's customized homepage is displayed on the client browser.
- **Response Measure**:
  - The average availability of the authorization service is increased to at least 99% per week.

Key Points

- Defining and applying scenarios ensures that desired quality attributes are shown.
- Scenarios define how the system responds to factors that affect quality properties.
- Should include the initial system state and environment state, external stimulus, required system response, and how to assess response.

Next Time

- Testing Fundamentals
- Exercise session: Scenarios

- **Assignment 1 due Feb 6.**
  - **Will be posted after team formation complete.**

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY